

A COMPACT RELATIONAL STRUCTURE REPRESENTATION

M. Shneier
Department of Artificial Intelligence
University of Edinburgh
Edinburgh, Scotland

Current Address:
Computer Science Center
University of Maryland
College Park, MD 20742

This paper is concerned with the representation of objects by computer. It presents a compact relational structure representation derived by generalizing the properties of objects. Objects are described by means of surface primitives (basic descriptive elements) and relations between the primitives. The essence of the representation is the use of a single primitive to describe all identical parts. The representation involves abstracting properties of individual scene elements. Similarly, relations between parts of objects that hold for particular instances of the objects are generalized to ensure that they will be valid for all instances. Common primitives and relations are shared both within and across models. This leads to a compact representation that has many desirable properties. A computer implementation of a version of the representation has been developed. It is used to illustrate the descriptive power of the representation. A recognition system has been developed that takes advantage of the representation in its operation. It is described elsewhere (Shneier (1977), Shneier (1978)).

1. INTRODUCTION

The way in which information is represented is critically important to the effectiveness of any system in Artificial Intelligence. Using a good representation, tasks that would otherwise be intractable become amenable to solution. In machine vision, representations have usually been tailored to specific, small domains. Recognition algorithms have been developed that depend for their effectiveness on specific knowledge about the objects in the domain. This paper presents a new representation for describing objects. The system is intended to be generally applicable, but was developed primarily for the domain of three-dimensional machine vision. It has also been applied to a second domain, that of spelling correction. The presentation of the system will be in terms of machine vision.

The representation takes advantage of similarities in the descriptions, or models, of objects. If two parts of an object, or two parts of different objects, have the same description, then these parts are represented by a single description in the representation. The basic idea is to share descriptions that are common within models and across models. When this sharing is accompanied by a comprehensive

indexing scheme, it leads to a representation that has many advantages over previous representations (e.g. Barrow and Popplestone (1971), Falk (1972), Ambler et al. (1975)) and suffers few disadvantages.

Some constraints were placed on the implementation of the system. The first constraint was that the descriptions of objects should be acquired automatically from visual data. This is a sensible constraint because it allows the domain to be changed easily and confronts the problems of representing objects from the real world, rather than those of representing artificially-created objects. The second constraint was that the representation should not be domain-dependent. It should be possible to change or expand the domain with as little effort as possible, and the information known about objects should be acquired entirely from the objects themselves rather than being partly dependent on built-in assumptions about the domain. This constraint would be useful in a robot assembly domain, where different assembly tasks might use widely different parts. Instead of having to re-program the system for each new task, it would be preferable for it to acquire descriptions of the parts visually.

Some of the problems to be confronted when a

representation and a recognition system are being designed become amenable to solution in a more natural way using the new representation than using earlier representations. It becomes easier to compare descriptions of objects, the domain of objects may be changed or expanded with very little effort, and the problems of matching symmetrical objects disappear. There is a great deal of flexibility in the choice of how to describe objects, and the form of the representation makes recognition of objects easier.

A recognition algorithm based on the representation has been developed and implemented. It makes use of the form of the representation to enable it to recognize objects in a robust and efficient manner. An earlier version of the algorithm appears in Shneier (1977). It is discussed in more detail in Shneier (1978), and examples are presented of its use in three-dimensional vision and in spelling correction.

2. COMPACT MODELS

The approach taken here differs from earlier work in two major respects. The first difference concerns the nature of models. Those authors who have previously made use of models of objects (e.g. Falk (1972), Barrow and Poplestone (1971), but not Waltz (1975)) have given the term "model" certain implicit characteristics. A model has been defined as a description of an object with a one-to-one relationship between parts of the object (e.g. edges, surfaces, volumes) and parts of the model. For example, the model for a cube consists of a separate description of each of the six sides of the cube, and of relationships between the sides.

There are a number of reasons why this concept of model, while natural, is not the best for machine vision. It means that each model has to be treated as a separate entity because there is no sharing of structure between models or parts of models. Most systems have performed matching separately on each model in turn, with models being rejected one by one until a match was found. It was desired to be able to match with all possible models in parallel, finding the best match in the most efficient manner possible. Grape (1973) attempted to match in this way by compiling a central list of features against which to match. He was not entirely successful, however, because he had to match against each element of the feature list in turn.

Another reason for re-examining the notion of model concerns the analysis of models of symmetric objects. Underwood and Coates (1972) were unable to deal with symmetric objects because of problems of finding the correct correspondence between parts of the scene and parts of the model. Other authors have had to content themselves with finding all possible correspondences between the scene and a model, and then choosing one of these arbitrarily (e.g. Barrow and Poplestone (1971), Ambler et al. (1975)). The problem arises because the model of a symmetrical object contains several parts that are described in a very similar fashion. When an object model is being constructed from several views, it is not clear when to construct a new part in the model and when to assume that all parts have already been seen. Similarly, when trying to recognize an object, it is not clear which of a number of parts of a model should be matched against a scene fragment. If the system is to work in a world of man-made objects, it seems advisable to be able to handle symmetry in a more acceptable manner.

Our solution to these problems with models is to broaden the concept of a model. A model is defined as a collection of descriptions of the kind of parts that make up an object and the set of the relations between them. There is no requirement of a one-to-one correspondence between object parts and model parts. Rather, there is one model part for each distinguishably different object part. This is similar to systems of mathematical logic, where a model is a collection of axioms describing a world, and where no axioms are repeated.

So, for example, a model of a cube consists of only one part description--that of a square plane. Relations are defined between instances of the part, so that a form of relational structure is the basis of the representation.

Figure 1 depicts the difference between the conventional relational structure model of a cube and the model described in this paper. The conventional model has one node for each surface in the cube, whereas the new model has a single node describing all the surfaces. Similarly, while there are many relations defined between the surfaces in the conventional model, the new model needs only a single relation to represent them all. There is a great saving in the size of the model.

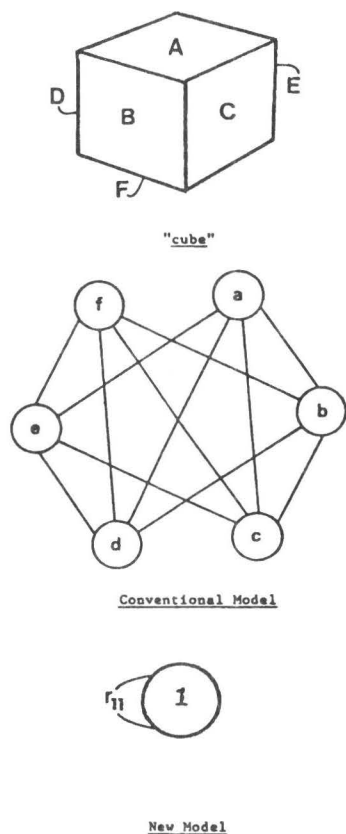


Figure 1

In fact, the models are even more like those of the logician than has been described so far. A logician describes a whole world with one set of axioms. Similarly, the models in this representation all share one structure, instead of the usual separation of models of different objects. The advantages of such a system are manifold. Matching is enormously simplified and matches occur with all possible models at once. Models are compact and are easily compared. Similarities are evident in shared structure, and differences are reflected by structure that is not shared.

The second difference between the new system and earlier work is more specific, and arises as a result of the formulation of the models as relational structures. A relational structure is defined as a set E of elements, together with a set of properties P and a set of relations R over E (Ambler et al, 1975)). Scene

analysis based on relational structures has been concerned with the following kind of matching: Given two relational structures $\langle E_1, P, R \rangle$ and $\langle E_2, P, R \rangle$ define a match between them as a set $T_1 \subseteq E_1$, a set $T_2 \subseteq E_2$, and an isomorphism between T_1 and T_2 preserving properties and relations. That is, $e_1 \sim e_2$ implies $p(e_1)$ iff $p(e_2)$ for each $p \in P$. Also, $e_1 \sim e'_1$ and $e_2 \sim e'_2$ implies $r(e_1, e'_1)$ iff $r(e_2, e'_2)$ for each $r \in R$. A match represents a common substructure in the relational structures.

This definition will be changed to allow the two relational structures to have a different relationship. For recognition, the structure of models and the structure obtained from the scene being recognized are described in terms of different basic elements. That is, the structure obtained from the scene (called the scene graph) represents the parts of the scene as they actually appear, whereas the structure of models (the graph of models) represents a meta-description of the objects possibly in the scene. Several parts of the scene graph may map into a single node of the graph of models. There is an isomorphism from the scene graph to a structure derived from the graph of models, but not to the graph of models itself. The structure derived from the graph of models is a conventional relational structure with each part of the object described individually, even if the parts have similar descriptions. Relations between the parts are also made explicit.

Thus it is necessary to distinguish three levels in the representation. There is the representation of the actual sense data in the scene graph, there is the structure representing the models of the objects known to the system, and in between there is a structure instantiated from the graph of models to aid in the analysis of the particular scene being studied. The relationship described above for matching relational structures holds between the scene graph and the intermediate structure, but this intermediate structure is only one of many possible such structures that can be instantiated from the graph of models.

Winston (1975), with his TYPICAL-MEMBER labels, went some way towards this position. He did not explore the potential of such a representation, but relied on the conventional structure representation.

3. REPRESENTATION

A representation has been implemented that is simple to construct and is powerful enough to describe a large class of objects. The representation is based on primitives and relations. Ideally, primitives could be descriptions of any scene elements that are easy to extract, have a canonical description, and have sufficient variation to allow the representation of an interesting class of objects. A small set of easily calculated relations is desired that succinctly describe the relationships between instances of the primitives in an object.

Relations are stored as schemata; their arguments are abbreviated surface descriptions instead of full surface descriptions. The form of the representation is a network whose nodes are primitives, and in which an arc is drawn between nodes if a relation is defined between the primitives at those nodes. Arcs are labelled with the set of relation schemata that justifies them. An object will be described by some set of these nodes and arcs, but many nodes and arcs in the network may be common to several object models. A primitive is represented by a single node, so should an instance of a primitive occur in several objects, the models produced from these objects would share that node. In addition, models may share relations, should common primitives occur in similar relationships in both models. The only thing that two models may not share is a name.

The network for the two objects of Figure 2 is given in Figure 3. The objects are a cube (called "cube"), and a rectangular parallelepiped ("rect") whose model shares a side with the cube.

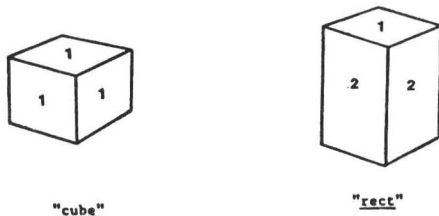
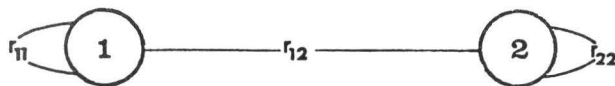


Figure 2



$r_{11} = \{ \langle \text{RELANGLE } P1 \ P1 \ 90^\circ \rangle, \langle \text{RELANGLE } P1 \ P1 \ 180^\circ \rangle \}$

$r_{12} = \{ \langle \text{RELANGLE } P1 \ P2 \ 90^\circ \rangle \}$

$r_{22} = \{ \langle \text{RELANGLE } P2 \ P2 \ 90^\circ \rangle, \langle \text{RELANGLE } P2 \ P2 \ 180^\circ \rangle \}$

Figure 3

Definitions

The definitions are of the terms used to describe the representation. They are in terms of labelled graphs.

1. Let $P = \{p_1, p_2, \dots, p_n\}$ be a set of primitives. Each p_i forms one (and only one) node of the graph.
2. Let $R = \{r_1, r_2, \dots, r_m\}$ be a set of relation schemata. Each r_i is of the form: $\langle \text{function name}, \text{arg}_1, \text{arg}_2, \dots, \text{arg}_n, \text{value} \rangle$. The arguments $\text{arg}_1, \text{arg}_2, \dots, \text{arg}_n$ are typed variables which may take as values instances of individual elements of P (possibly repeated). The value FALSE (0) is not considered legal for a relational schema. The labels on the arcs of the graph are sets of relation schemata, subsets of R , called label sets.
3. Let $M = \{n_1, n_2, \dots, n_k\}$ be a set of names of models.
4. The graph obtained from modelling objects is called the graph of models.
5. A model in the system is represented by a subgraph of the graph of models. The nodes in the subgraph are those primitives that occur in the model. The arcs in the subgraph are those that have relation schemata in their label sets that occur in the model. The label set of an arc in the subgraph consists of the subset of relations that occur in the model.

A rich indexing structure is maintained by the representation. Each primitive indexes the models in which it occurs. If a primitive forms part of the description of an object, it must index the model for that object.

Relation schemata index the models in which they occur and the primitives that form their arguments. To test whether or not a relation is satisfied requires finding instances of its arguments, while to derive any benefit from such a test requires that the information obtained be available to the models that can use it.

The models index both the primitives that form their parts and the relation schemata that describe how the parts are related.

4. THE MODELLING PROCESS

4.1 Extraction of surfaces

The implementation discussed here makes use of range data obtained from a triangulation ranging system (Popplestone et al. (1975)). An object to be modelled is placed on a turntable (Figure 4). The object is passed under a plane of light, and scanned at fixed intervals. When the scan is complete, the object is rotated by a known amount, and scanned again. This process is repeated until all desired views of the object have been scanned (Figure 5).

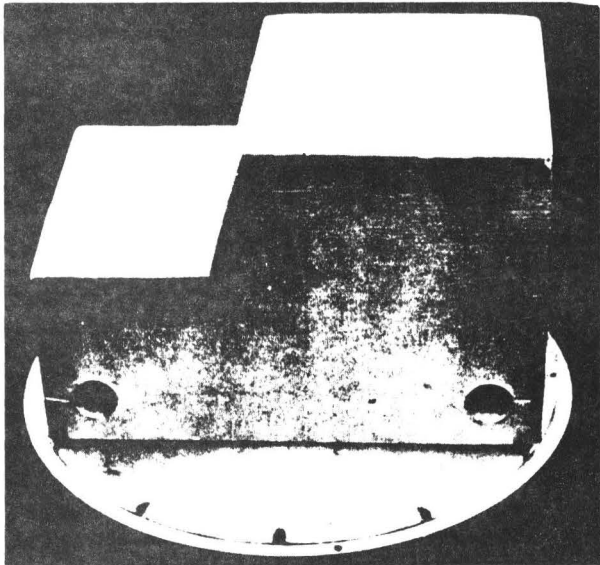


Figure 4

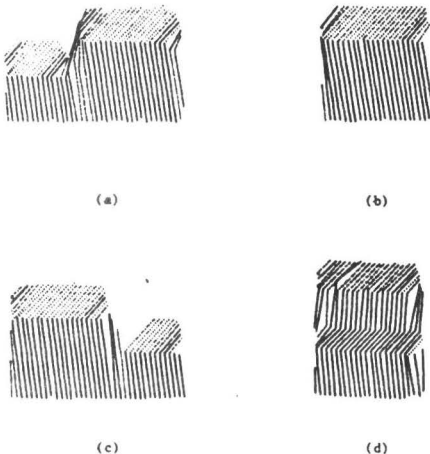


Figure 5

The result of processing the data obtained in this way is a set S of surfaces, each described by:

1. The surface type (plane or cylinder).
2. The equation of the surface.
3. The three-dimensional coordinates of the centroid of the surface.
4. The radius of a cylindrical surface.
5. The segments of lines that make up the surface.

Some further processing is done on this information before a model is produced. The aim of this processing is to derive simple, non-coordinate-specific information about the surface, which can be used to form primitives. The information chosen is a measure of the dimensions of the surface.

4.2 Primitives and relations

From the set S of surfaces, a list P' of surface descriptions is constructed, one for each surface. Elements of P' may be repeated. The elements of P' will be used to construct the set P of nodes in the graph of models. The descriptions are constructed from the surfaces by extracting the type of surface and its dimensions. All coordinate-specific information is discarded when forming the descriptions. This information is, however, still available to the rest of the system, and will be used to construct the relation schemata. At the same time, the set

$$SP = \{ \langle s, p \rangle \mid s \in S, p \in P \}$$

of pairs of surfaces and their associated descriptions is constructed. SP links the surfaces in the image with the descriptions in the model world. The three-dimensional positions of the surfaces and their relationships with other surfaces will be used in building the model.

The next step is to determine the relationships between the surfaces in the object being modelled. In the implementation, relations have been restricted to having at most two arguments, for ease of acquisition and because the number of relation schemata produced for n -ary relations can be very large. In practice, binary relations are adequate for the domains under consideration, although Shneier (1978) presents an argument to show that this is not true in general.

The choice of the relations with which to model an object is left to the user, who lists the names of the functions to be applied at the time the model is constructed. The use of a relation requires that an executable function exist which can calculate values for instances of its arguments. A number of different functions have been used in experiments. Some of these are:

ADJACENT: If the light plane in some subscan passed over both surfaces at the same time, and the line segments produced by the light plane were touching, then the surfaces are adjacent.

RELDIST: The distance from the centroid of one surface to that of another.

RELANGLE: The angle between two surfaces, from the viewpoint of the first.

Whereas ADJACENT is a true relation, in that it has values "true" or "false," RELDIST and RELANGLE can have a large number of values, each of which will give rise to a new schema when it is encountered in an object being modelled. (There will not, however, be an infinite number of such values, since a range of values which are sufficiently similar will be grouped into an equivalence class.)

The schemata are constructed as follows. For each pair of surfaces in S, the chosen set of functions is applied. The result of the application is used as the expected value slot in the schema. The name of the function that was applied is the value of the function name slot, while the argument slots are filled by the descriptions that correspond to the surfaces used in the evaluation. The correspondence is obtained from the set SP of pairs of surfaces and their descriptions. Using abbreviated descriptions, rather than full surface descriptions, generalizes the relations to make them valid for all surfaces that match the abbreviated descriptions. If the value of the function is FALSE (0), a schema is not constructed.

4.3 Producing the graph of models

At this stage there is

1. A list P' of surface descriptions.
2. A set of relation schemata.
3. A model name.

This information must be integrated into the graph of models. Any new information should

be retained, but duplication or spurious arcs should not be introduced.

Each of the descriptions in P' is matched against the primitives in P, the set of nodes of the graph of models. Only if there is no match (i.e. if a description is sufficiently different) is a new primitive node created for the description and added to P. Primitives are described in exactly the same way as surface descriptions, but there is only one primitive with a particular description. The result is a set PUP' of primitives. Matching requires that the surface types be the same in the primitives and the surface descriptions, and that the dimensions be within some fixed tolerance of each other (currently, dimensions are allowed to vary by about 20% before the match fails).

When a surface description matches with a primitive already in the graph, the values of the primitive in the graph are updated. The mean of each description in the surface and the primitive becomes the dimension for the primitive in the graph.

The name of the model is added to the set of names if it is not already a member. The new set of relation schemata is added to the set R of schemata that already exist as follows. If a new schema is not already a member of R, it is added to R, and indexes only the new model. If a relation scheme already exists in R, it is not duplicated. Instead, the schema is altered so that it indexes the new model in addition to any others it might already index. The model indexes the primitives and relation schemata that occur in it, and the primitives index the new model.

4.4 Example

Figure 4 shows a picture of a toy car resting on the turntable. The car was scanned by the ranger, rotated through 90 degrees, and scanned again. This process was repeated four times, giving four views, each perpendicular to its predecessor. The four views cover all surfaces of the car except its base (Figure 5).

After scanning the car, the range map produced from the scans is processed to find the surfaces of the car. There are seven of these, corresponding to the roof, two sides, back, front, windshield, and bonnet of the car. Some of these surfaces were visible in more than one view of the car. The roof appeared in all four scans, and the bonnet in three (Figure 5). Notice that the sides of the car appear smooth

because the axle holes were not large enough to be perceived by the ranger.

From the set of surfaces, the program must produce a description of the car. To be able to do this, three pieces of information are necessary. The first is a name for the object--it will be called "CAR". The second requirement is a file containing the surface data, and the third requirement is a list of functions to be applied to pairs of surfaces to produce the relation schemata. In this example, a single function that finds ADJACENT relationships between surfaces will be used.

The first thing that the program does is to work out descriptions for each surface. It finds the type of the surface (in this example, all surfaces are planes) and two numbers that describe the dimensions of the surface.

When the program tries to find the dimensions of the roof of the car, it discovers that there are several views of the surface, that is, the roof was scanned more than once by the ranger. Each of these views is processed separately, and the results are merged. The bonnet of the car is treated similarly.

Having described the surfaces, the next stage is to construct the relational structure to describe the object. First, the program examines the surface description. If two descriptions are similar, they are represented by a single primitive node (PRIMITIVE) in the graph of models. In the example, three primitives are found to be sufficient to describe the seven surfaces. This is less than half the number that would be required by a conventional representation, illustrating the compactness of the representation. The three primitives describe the following surfaces:

Primitive P1 describes the roof and back of the car.

Primitive P2 describes the windshield, bonnet, and front of the car.

Primitive P3 describes the two sides of the car.

The dimensions stored for each of the primitives are the means of the dimensions of each surface that matched with the primitive. In addition to describing the surfaces, the primitives also index the model CAR of which they are parts.

The next step in building the model is to link the primitive nodes by means of ADJACENT relation schemata. An arc is constructed between two nodes if any of the surfaces described

by one of the nodes is adjacent to any of the surfaces described by the other node. Thus, the side of the car is adjacent to the back of the car, giving rise to the schema <ADJACENT P1 P3 >. The side of the car is also adjacent to the roof of the car, but this gives rise to the same schema, because the roof and the back are described by the same primitive and one schema is used to represent both relationships. All pairs of surfaces are examined, and five relation schemata are found sufficient to describe the relationships between the surfaces in the car.

The three primitives and five relation schemata, together with the name of the car, are all the information needed to model the car (Figure 6).

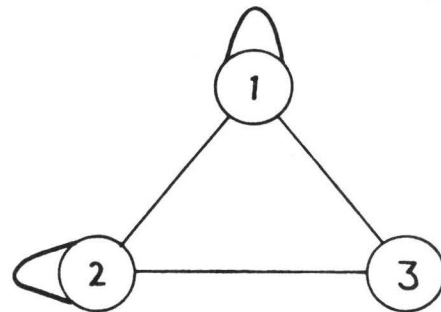


Figure 6

Note that the bottom of the car was not scanned, and so does not appear in the model. To incorporate it, two sets of scans would be necessary. The model as shown would first be constructed, and then another set of views of the car, this time on its side or its roof, would be produced. Presenting the modeller with the new set of views and the old object name would result in the model for CAR being updated to include the new surface and any new relations that were discovered.

The situation is more complex when several objects are modelled. If the descriptions of some of the surfaces of an object being modelled match with primitives already in the graph of models, the new model will share that primitive in the graph. The node is made to index the new model in addition to any others it already indexes. Similarly, relation schemata may be shared by several models, and must be indexed accordingly.

5. DISCUSSION

The advantages of the representation are as follows. The representation is compact because nodes are shared by parts of several objects. This allows the space required to store models to be reduced, and has valuable implications for recognition. Models can be constructed easily

from visual data, and new models can be integrated into the network without affecting those models already in existence.

Because of the sharing in the structure, all models that are described by a part of the structure are made available simultaneously when that part of the structure is accessed. Instead of matching with each model individually, all of them are available at once and the work that is done to discriminate between models can be tailored by the context. This allows less work to be done when a scene is analyzed.

The representation is flexible. It was easy to change the domain to which the system was applied from three-dimensional vision to one-dimensional spelling correction without changing the form of the representation. It is also possible to describe objects within a domain in different terms.

The ability to compare descriptions is one of the criteria for a useful recognition. In the past, objects have each been described separately and compared by finding correspondences between two structures. In the representation presented here, a single structure is shared by all models, with parts of models that are similar being represented by the same part of the structure. Thus, instead of painfully comparing two structures and finding points of correspondence, the single structure can be examined and similarities and differences between object models can be found immediately. Similarities are indicated by shared structure, and differences are indicated by structure that is not shared.

An important requirement for a representation and recognition system, especially one that is to work with man-made objects, is an ability to deal with symmetry. Many objects have some degree of symmetry, and this has led to problems in previous systems. The difficulty arises when a symmetrical object is matched with its model description. It is not clear what parts of the object should be matched with what parts of the model because there is no unique way of matching when several parts have the same description. This problem has either been avoided by not allowing symmetric objects in the domain (Underwood and Coates (1972)), or it has been solved by resorting to finding all possible correspondences between the object and its model and choosing one of them (e.g., Barrow et al. (1972)). When there is a large amount of symmetry, as in a cube, this method

can be very expensive. Using the representation in this paper, the problem of symmetry disappears. There is a single description for all the similar parts, so that there is no confusion when matching. Symmetric objects can thus be recognized with the same effort as non-symmetric objects.

The representation has been successfully applied to two domains. The first, machine vision, used surface descriptions as primitives, and relations like adjacency. The second, spelling correction, used letters as primitives and ordering relations like "precedes" and "follows". Shneier (1978) provides a full discussion of each domain. He also discusses an elegant recognition algorithm that takes advantage of the form of the representation in its analysis. The modelling and recognition systems have been fully implemented and tested. Their performance underlines the advantages of the compact representation.

6. CONCLUSIONS

This paper has presented a new representation that takes advantage of similarities in the descriptions of objects. By sharing descriptions that are common within models and across models, a compact representation was obtained that had numerous advantages over other representations. The system was implemented and applied to two domains, those of machine vision and spelling correction. In both cases, it proved possible to represent the objects in the domain in a compact manner that was especially useful for recognition. The simplification of the representation is a direct result of a generalized concept of model.

ACKNOWLEDGEMENTS

I am grateful to my colleagues in the Department of Artificial Intelligence at the University of Edinburgh for their assistance in this work, to Azriel Rosenfeld for his comments and suggestions on the form of this paper, and to the University of Witwatersrand who provided financial support.

REFERENCES

- A. P. Ambler, H. G. Barrow, C. M. Brown, R. M. Burstall, and R. J. Popplestone (1975): A Versatile System for Computer-Controlled Assembly. Artificial Intelligence, Vol. 6, No. 2, pp. 129-156.
- H. G. Barrow, A. P. Ambler, and R. M. Burstall (1972): Some Techniques for Recognizing

Structure in Pictures. In: Frontiers of Pattern Recognition. Ed. Watanabe. New York, Academic Press.

H. G. Barrow and R. J. Popplestone (1971): Relational Descriptions in Picture Processing. Machine Intelligence 6. Eds. B. Meltzer and D. Michie. Edinburgh University Press, pp. 377-396.

G. Falk (1972): Interpretation of Line Data as a Three-Dimensional Scene. Artificial Intelligence, Vol. 3, No. 2, pp. 101-144.

G. R. Grape (1973): Model Based (Intermediate Level) Computer Vision. Ph.D. Thesis, Department of Computer Science, Stanford University.

R. J. Popplestone, C. M. Brown, A. P. Ambler, and G. F. Crawford (1975): Forming Models of Plane- and Cylinder-Faceted Bodies from Light Stripes. Proc. 4th IJCAI, Tblisi, pp. 664-668.

M. O. Shneier (1977): Recognition Using Semantic Constraints. Proc. 5th IJCAI, Cambridge, Mass., pp. 585-589.

M. O. Shneier (1978): Object Representation and Recognition in Machine Vision. Ph.D. Thesis, Department of Artificial Intelligence, University of Edinburgh.

S. A. Underwood and C. L. Coates (1972): Visual Learning and Recognition by Computer. TR 123, Information Systems Research Laboratory, University of Texas at Austin.

D. Waltz (1975): Understanding Line-Drawings of Scenes with Shadows. In: The Psychology of Computer Vision, Ed. P. H. Winston. New York, McGraw-Hill.

P. H. Winston (1975): Learning Structural Descriptions from Examples. In: The Psychology of Computer Vision, Ed. P. H. Winston. New York, McGraw-Hill.